

# Technical Cybersecurity

Formatting your stack

# Complexity++

## STACK FORMATTING IS MUCH MORE DIFFICULT

---

- ▶ We need to move values into registers
  - ▶ 0x0b is the syscall number for **execve(.)**, must be in al
  - ▶ ebx must point to the string **execve(.)** will execute
  - ▶ ecx must point to the argv pointer array
  - ▶ edx must point to the envp array

# Stack to Generate

## VALUES & OPCODES

---

- ▶ You need to intersperse data with opcodes
- ▶ Data gets used during interpretation to create special byte arrangements

**-> xor eax eax**

**-> pop ecx ; pop edx**

0x0b0b0b0b

(pointer to NULL BYTE)

**-> mov [edx+0x18] eax**

**-> or al cl**

**-> pop ebx**

(pointer to “/bin/sh”)

**-> pop ecx; pop edx**

(pointer to argv)

(pointer to env vars)

**-> int 0x80**

(pointer to “/bin/sh”)

*NULL BYTE*

*“/bin/sh”*

# From here

## PYTHON PROGRAM

---

- ▶ Create a python program to generate your stack to inject
  - ▶ We'd use the same approach, but at a much larger scale
  - ▶ Create, first, with recognizable bytes
- ▶ Send it to a file
- ▶ Read into the file at runtime
  - ▶ (gdb) r filename
  - ▶ \$ rop filename
- ▶ Read and debug resulting core files

Modern exploitation is  
difficult!